(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
12 August 2010 (12.08.2010)

PCT

(10) International Publication Number
WO 2010/090745 A1

(51) **International Patent Classification**:
G06F 12/08 (2006.01) *G06F 3/06* (2006.01)

(21) **International Application Number**:
PCT/US2010/000317

(22) **International Filing Date**:
4 February 2010 (04.02.2010)

(25) **Filing Language**: English

(26) **Publication Language**: English

(30) **Priority Data**:
61/150,380 6 February 2009 (06.02.2009) US

(71) **Applicant** *(for all designated States except US)*: **OSR OPEN SYSTEMS RESOURCES, INC.** [US/US]; 105 State Route 101A, Suite 19, Amherst, NH 03031 (US).

(72) **Inventors; and**
(75) **Inventors/Applicants** *(for US only)*: **MASON, W., Anthony** [US/CA]; 1203-821 Cambie Street, Vancouver, British Columbia V6B 0E3 (CA). **WIDDOWSON, Roderick, David, Wolfe** [GB/GB]; The Steading, Newmains, Stenton, East Lothian, Scotland EH42 1TQ (GB).

(74) **Agents**: **WOLFE, Christopher, G.** et al.; K&L Gates LLP, Henry W. Oliver Building, 535 Smithfield Street, Pittsburgh, PA 15222-2312 (US).

(81) **Designated States** *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) **Designated States** *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

— *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii))*

[Continued on next page]

(54) **Title**: METHODS AND SYSTEMS FOR DATA STORAGE

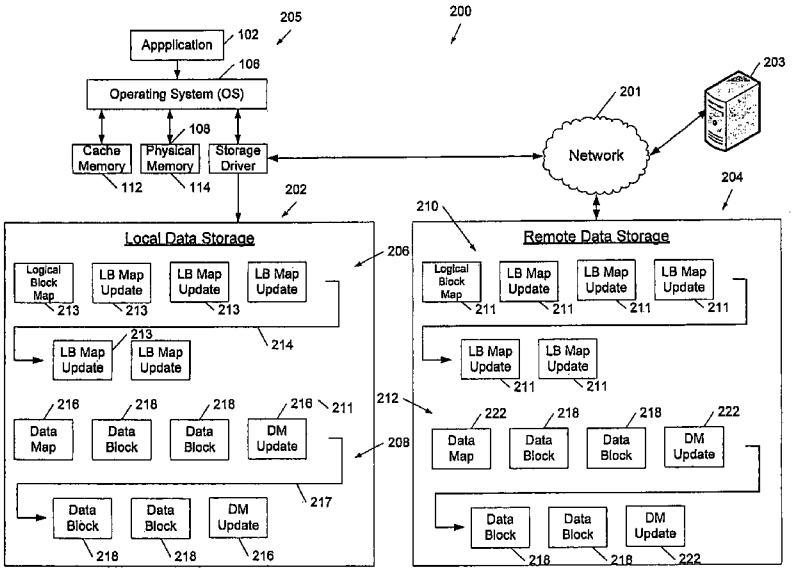

Figure 2

(57) **Abstract**: In one general aspect, various embodiments are directed to a method of writing a data block to a memory comprising receiving an electronic write request from an application. A content address of a first data block considering the value for the first data block. A mapping of the first data block to the content address may be written to a logical end of the local block map. The mapping may also be written to a remote block map. If the content address is not present at a local data storage, the value of the first data block may be written to the local data storage at a first location and metadata associating the content address with the first location may be written to the local data storage.

WO 2010/090745 A1

Published:

— *with international search report (Art. 21(3))*

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

METHODS AND SYSTEMS FOR DATA STORAGE

W. Anthony Mason

Roderick David Wolfe Widdowson

PRIORITY CLAIM

[0001] This application claims the benefit of U.S. Provisional Patent Application 61/150,380 filed on February 6, 2009, which is incorporated herein by reference in its entirety.

BACKGROUND

Increasing numbers of computer devices utilize remote, server-side data storage, including web-based data storage. According to typical server-side storage arrangements, a service provider implements one or more network-accessible hosts. Each host usually comprises data storage hardware and one or more servers for administering the hardware. The hosts and data storage hardware may be at a single physical location, or may be distributed across multiple location. Users of the service are able to access the hosts over the network to upload and download data files. The network may be a local area network (LAN) or a wide area network (WAN), such as the Internet. Typically, the users can access the central data store from multiple computer devices, and often from any computer device having the appropriate client software and the ability to communicate on the network. The service provider may be a private enterprise providing data storage to its employees and other affiliates. Also, the service provider may be a commercial entity selling access to its storage. One example of such a commercially available remote data service is the SIMPLE STORAGE SERVICE or S3 available from AMAZON WEB SERVICES LLC.

Remote or server-side data storage has a number of advantages. For example, remote data storage is often used as a means to back-up data from client computers. Data back-up, however, is only effective if it is actually practiced. Backing up files to a remote data storage can be a tedious and time consuming task that many computer users just do not do. As more individuals store important information on their mobile telephones and personal digital assistants (PDA's), backing up these devices is becoming prudent as well.

BRIEF DESCRIPTION OF THE FIGURES

Various embodiments of the present invention are described here by way of example in conjunction with the following figures, wherein:

Figure 1 shows a block diagram of one embodiment of a client system architecture.

Figure 2 shows a block diagram of one embodiment of a system comprising a client device organized according to the architecture of Figure 1 and utilizing a local data storage and a remote data storage as a component of its data storage.

Figure 3 illustrates one embodiment of a process flow for writing data blocks to data storage in the system of Figure 2.

Figure 4 illustrates one embodiment of a process flow for reading a data block using the system of Figure 2.

DESCRIPTION

Various embodiments are directed to systems and methods for implementing content addressable, log structured data storage schemes, which may be implemented on a single machine or across multiple machines as part of a remote storage system. In some embodiments, content addressable, log structured data storage may be used to allow client devices to utilize remote storage as their primary, bootable data storage and/or may facilitate data back-up utilizing remote data storage. In embodiments where the remote storage is used as a client's primary data storage, data may be cached at local storage, but ultimately pushed to the remote storage. In this way, valuable user data may be concentrated at the remote data source, allowing for easier data management, updating and back-up.

In various embodiments, the content addressable, log-structured nature of the data storage schemes may address existing shortcomings of remote data storage that currently make it undesirable for use as a bootable primary data storage. On such shortcoming is related to access times. Access times for remote storage are often greater than access times for local storage. On the pull or download side, a client machine may achieve acceptable access times and minimize pulls from the remote storage by locally caching data that is subject to repeated use. Further,

many implementations of remote storage are configured to minimize pull times, which may increase the effectiveness of caching or even make it unnecessary.

The optimization of remote data source pull times, though, often comes at the expense of longer push times. Push times on some commercially available remote storage solutions can be between several seconds and several hours. Accordingly, it may be desirable to minimize data being pushed to remote storage. In various implementations, the content addressable, log-structured data storage described herein may address this concern. Because the data storage is content addressable, the client may not have to push a new data block if a data block with the equivalent content already exists at the remote data source. Because the data storage is log-structured, writing to or modifying the remote storage may only require pushing a new data block, if any, and pushing short modifications to one or more logs describing the new data block. Although the content addressable, log-structured data storage has certain disclosed advantages when used in a remote storage environment, it may also be used to achieve other advantages, for example, on a single machine.

Figure 1 shows a block diagram of one embodiment of a client system architecture 100 comprising content addressable, log-structured data storage 110. The architecture 100 may be implemented by a client computing device in a remote storage environment. For example, the data storage 110 may comprise local and remote data storage portions. The data storage 110 and the various components of the architecture 100 may be implemented utilizing software and/or hardware. For example, in addition to the data storage 110, the architecture 100 may comprise one or more examples of an application 102, an operating system 106, a storage driver 108, cache memory 112, physical memory 114 as well as other common components that are not shown.

The application 102 may include a group of one or more software components executed by a processor or processors of the client device. It will be appreciated that the architecture 100 may, in various aspects, include additional applications (not shown) that may execute sequentially or simultaneously relative to the application 102. The application 102 may perform at least one task such as, for example, providing e-mail service, providing word processing, providing financial management services, *etc.* Applications, such as the application 102 may perform tasks by manipulating data, which may be retrieved from the data storage 110 and/or memory 112, 114.

Interaction between the application 102 and the data storage 110 and memory 112, 114 may be facilitated by the operating system 106 and the storage driver 108. The operating system 106 may be any suitable operating system. For example, in various non-limiting embodiments, the operating system 106 may be any version of MICROSOFT WINDOWS, any UNIX operating system, any Linux operating system, OS/2, any version of Mac OS, *etc.* To acquire data for manipulation and output results, applications 102 may generate "read requests" and "write requests" for particular data blocks.

A data block may represent the smallest unit of data handled by the architecture 100 and/or stored at data storage 110. Logical constructs, such as files, may be expressed as one or more data blocks. Metadata may also be expressed as one or more data blocks. Data blocks may be of any suitable size, depending on the implementation of the client system 100. For example, many physical storage drives have disks with sectors that are 512 bytes. Some disks may have 520 byte sectors, leaving 512 bytes for data and 8 bytes for a checksum. Other disks, such as some SCSI disks, may have 1024 byte data blocks. Accordingly, some embodiments may utilize data blocks that are 512, 520 and/or 1024 bytes in size. Also, for example, a typical file system sector may be 4096 bytes or 4 kilobytes (kB) and, some physical storage devices, such as CD-ROM's, have sectors that are 2048 bytes (2 kB). Accordingly, 4 kB and 2 kB data blocks may be desirable in some embodiments.

The read and write requests originating from the application 102 are provided to the operating system 106. (It will be appreciated that some read and write requests may originate directly from the operating system 106.) In various embodiments, the application 102 may utilize an application program interface (API) or other library (not shown) to facilitate communication between the application 102 and the operating system 106. The operating system 106 may service read or write requests from the application 102, for example, by accessing data storage 110 through the storage driver 108, or by accessing memory 114, 112. Physical memory 114 (*e.g.*, Random Access Memory or RAM) may include volatile or non-volatile memory with read and write times that are faster than those of the data storage 110. The operating system 106 may utilize physical memory 114 to store data that is very commonly read or written to during normal operation, thus reducing access times and increasing execution speed. Accordingly, some read or write requests from the application 102 may be handled directly from memory 112, 114. Optional cache memory 112 may be faster than physical memory 114 and may be used for a similar purpose.

Many read and write requests, however, require the operating system 106 to access data storage 110. In these instances, the operating system 106 may package read or write requests and provide them to the storage driver 108. Read requests provided to the storage driver 108 may comprise an identifier(s) of a data block or blocks to be read (*e.g.*, a logical block identifier). Write requests provided to the storage driver 108 may comprise identifier(s) of a data block or blocks to be written, along with the data blocks to be written. The storage driver 108 may execute the read and write requests. For example, in response to a read request, the storage driver 108 may return the requested data block or blocks. In response to a write request, the storage driver 108 may write the included data block. It will be appreciated that in various embodiments, some or all of the functionality of the storage driver 108 may be implemented by the operating system 106.

Physically, the data storage 110 may include any kind of storage drive or device capable of storing data in an electronic or other suitable computer-readable format. In some embodiments, data storage 110 may include a single fixed disk drive, an array of disk drives, an array of disk drives combined to provide the appearance of a larger, single disk drive, a solid state drive, *etc.* Data storage 110 may be local, accessible directly to the operating system 106, or may be remote, accessible over the network, such as the Internet. In various embodiments, the data storage 110 may comprise local and remote portions.

Logically, the data storage 110 may be implemented according to a content addressable, log-structured scheme. In a log-structured organization, data blocks and metadata describing the data blocks are written to a data source sequentially. To retrieve data blocks, the metadata is consulted to determine the location of the desired data block. In content addressable schemes, each data block is described by a representation of its content (*e.g.*, a content address). A content address for a block may be found, for example, by applying a hash algorithm to the data block. The hash algorithm may return a number, or hash, of a predetermined length. The hash represents the content of the data block. Depending on the quality of the hash algorithm used, it may be highly unlikely that two data blocks having different values will return the same content address or hash (*e.g.*, a collision). Example hash algorithms may include SHA-0, SHA-1, SHA-2, SHA-3, MD5, *etc.* Different algorithms, and different versions of each algorithm may yield hashes of different sizes. For example, the SHA-2 algorithm may yield hashes of 28, 32, 48, 64 bytes or more. The likelihood may be dependent on the quality of the hash algorithm, the length of the hash, and the size of the data block to be hashed. For example, when utilizing a larger

data block, it may be desirable in some circumstances to select a hash algorithm generating a longer hash.

Content addressable storage may utilize two layers of mappings. A logical block map, or block map, may link an identifier of a data block provided in a read or write request to a corresponding hash or content address. The identifier of the data block may be a name of a file or file portion, a disk offset, or other logical unit. A data mapping may map the hash or content address of a data block to the data block (*e.g.*, a physical location of the data block, or other way to access the data block). A read request received from the operating system 106 may comprise an identifier of the block or blocks to be read. The block map may be used to convert the identifier or identifiers to one or more hashes or content addresses. The data map may be used to return the identified data block or blocks given the hash or content address. A write request may comprise an identifier of and an indication of the value of a block (or blocks) to be written. The hash algorithm may be applied to the value to generate a content address. The content address may then be associated with the identifier in the block mapping. In a content addressable storage, it is possible for more than one identifier to correspond to the same content address and therefore to the same location in physical storage. For example, if two or more data blocks have the same value, only one instance of the data block may be stored at the data storage 110. Accordingly, if the content address and data block to be written are already stored at the data storage, there may be no need to re-write the data block. The block map, however, would be updated so that the identifier included in the request points to the exiting data block having the same content address.

According to various embodiments, the content addressable mapping functions may be implemented by the operating system 106, or the storage driver 108 of the architecture 100. In some embodiments where the mapping functions are implemented by the storage driver 108, their implementation may be transparent to the operating system 106 and the application 102. For example, the operating system 106 may provide disk offsets as identifiers for each data block in a read or write request. The storage driver 108 may implement the block mapping and the data mapping to return the data blocks to the operating system 106 and/or write the blocks to storage 110. In this way, the operating system 106 may believe that it is reading and writing from a local disk even if the data storage 110 comprises local and remote portions.

Figure 2 shows a block diagram of one embodiment of a system 200 comprising a client device 205 organized according to the architecture 100 and utilizing a local data storage 202 and a remote data storage 204 as a component of its data storage 110. Accordingly, the data storage

110 illustrated in Figure 1 may be embodied by a local data storage 202 and a remote data storage 204. The local and remote data storage 202, 204 shown in Figure 2 also illustrate a content addressable, log-structured implementation. The local data storage 202 may comprise any suitable kind of physical data storage device including, for example, a random access memory (RAM), a read only memory (ROM), a magnetic medium, such as a hard drive or floppy disk, an optical medium such as a CD or DVD-ROM or a flash memory card, *etc.* The remote data storage 204 may comprise any suitable kind data storage located remotely from the client 205. The remote data storage 204 may be accessible to the client via a network 201 such as, for example, the Internet. One or more servers 203 may administer the remote data storage 204. According to various embodiments, the remote data storage 204 may comprise a cloud storage system.

The local storage 202 may comprise a local logical block log, or local block log 206 and a local data log 208. The local block log 206 may comprise a local logical block map or local block map comprising local block map units 213. The local block map may implement the block mapping function of the data storage system. For example, the local block map may comprise a table or other data structure linking data block identifiers (*e.g.*, received from the operating system 106) with corresponding content addresses (*e.g.*, hashes). The units 213 making up the local block map may be written in sequential log-structured format. Units 213 indicating changes to the local block map may be written to the logical end of the log 206. For example, arrow 214 indicates the logical direction of updates. To find the current state of the local block map, the client system 205 (*e.g.*, via device driver 108) may either start at the logical beginning of the log 206 and consider each recorded change or start at the logical end of the log 206 and continue until the most recent change to the mapping of a desired data block is found.

The local data log 208 may comprise a data map units 216 and data blocks 218. The data map units 216 and data blocks 218 may be commingled in a log-structured format. It will be appreciated, however, that, in some embodiments, data blocks 218 may instead be commingled with the local block log 206 or may be included in a separate log (not shown). The data map units 216 may, collectively, make up a local data map which may map various content addresses to data units. Generally, the local data log may indicate which data blocks are cached at the local data storage 202. If a data block is not cached at the local data storage 202, then the client device 205 may retrieve the data block at the remote data storage, as described below.

The remote data source 204 may comprise a remote logical block log 210 and a remote data section 212. The remote block log 210 may comprise remote block log units 211, which

may be stored at the remote data source in a log-structured fashion. Collectively, the remote block log units 211 may make up a remote block log. The remote block log may be substantially similar to the local block log in most circumstances. That is, data block identifiers utilized by the operating system 106 should generally map to the same content address at the local block map and the remote block map. For example, the local block map may serve as a local cache copy of the remote block map. If the local block map is lost, it may be substantially replaced by pulling the remote block map.

The remote data section 212 may comprise data blocks 218, which may be organized in any suitable fashion. In the embodiment pictured in Figure 2, the data blocks 218 are organized in a log-structured fashion with remote data map units 222 making up a remote data map that describes the position of each data block in the log by its content address. Any other suitable method of indexing the data blocks 218 by content address may be used, however. For example, in various embodiments, the data blocks 218 may be stored hierarchically with each layer of the hierarchy corresponding to a portion of the content address (*e.g.*, the first x bits of the content address may specify a first hierarchy level, the second x bits may specify a second hierarchy level, and so on). Also, in other embodiments, the data blocks 218 may be stored according to a SQL database or other organization structure indexed by content address.

Figure 3 illustrates one embodiment of a process flow 300 for writing data blocks to data storage in the system 200. Although the process flow 300 is described in the context of a write request regarding a single data block, it will be appreciated that the steps could be easily modified for write requests comprising more than one data block. Referring to the process flow 300, a write request may be generated (302). The write request may include an identifier of a data block (*e.g.*, a disk offset) and a value for the data block. According to various embodiments, the write request may originate from an application 102, be formatted by the operating system 106 and forwarded to the storage driver 108. A hash algorithm may be applied to data block value included in the write request (*e.g.*, by the storage driver 108) to generate a content address (304). The storage driver 108 may update the local block map to associate the identifier with the content address corresponding to the data block value (306). This may be accomplished, for example, by writing a local block map unit 213 comprising the update to the end of the local block log 206. If the remote data storage 204 is available (308), then the remote block map may also be updated (310), for example, by pushing a remote block map unit 211 indicating the association to the end of the remote block log 210. If the remote data storage 204 is not available, the local block map unit 213 may be marked as un-pushed.

The storage driver 108 may traverse the local data map to determine if the content address is listed in the local data map (312). If the content address is not listed in the local data map, it may indicate that no existing data block on the local data storage 202 has the same value as the data block to be written. Accordingly, the data block to be written may be written to the end of the local data log 208 along with a local data map unit 216 mapping the content address of the data block to be written to its physical location in the log 208 (314). According to various embodiments, the local copy of the data block may be maintained, at least until the client 205 is able to verify that the data block has been correctly written to the remote data storage 204.

The storage driver 108 may also determine if a data block having the same content address as the data block to be written is present at the data section 212 of the remote data storage 204 (316). In embodiments where the data section 212 is log structured, this may involve traversing a remote data map comprising remote data map units 222. In embodiments where the data units are stored hierarchically at the data section 212, this may involve examining a portion of the hierarchy corresponding to the content address to determine if a value is present. In embodiments where the data units are stored in an indexed fashion (*e.g.*, at a SQL server), it may involved performing a search using the content address as an index. If no data block having the same content address as the data block to be written is present at the remote data storage 204, then the value of the data block to be written may be pushed to the remote data storage 204, if it is available. If the remote data storage is not available, then the local data log may be updated to indicate that the data block to be written has not be pushed to the local data storage 204.

The availability of the remote data storage 204 may, in various embodiments, depend on the network connectivity of the client 205. For example, when the client is able to communicate on the network 201, the remote data storage 204 may be available. It will be appreciated that when the client logs on to the network 201 after having been disconnected for one or more write requests, there may be one or more data blocks 218 and local block map units 213 that have not been pushed to the remote data storage 204. In this case, for each local block map unit 213 that is un-pushed, step 310 may be performed to update the remote block map. Likewise, for each data block 218 that is un-pushed, steps 316 and 318 may be performed to first determine if the data block 218 is present at the remote data storage 204 and, if not, push the data block 218 to the remote data storage 204.

Figure 4 illustrates one embodiment of a process flow 400 for reading a data block using the system 200. Although the process flow 400 is described in the context of a read request

regarding a single data block, it will be appreciated that the steps could be duplicated for read requests comprising more than one data block. Referring to the process flow 400, a read request may be generated (402). The read request may comprise an identifier for the data block to be read. If the identifier is listed in the local block map (404), then the local block map may be utilized to find the content address associated with the identifier (406). If the identifier is not listed in the local block map, then the remote block map may be used to find the content address associated with the identifier (408). If the identifier is not listed in the local block map, and the remote data storage 204 is not available, the read request may fail. After obtaining the content address corresponding to the requested data block, it may be determined if the content address appears in the local data map (410). If so, then the requested data block may be returned from local storage 202 (412). If not, then the requested data block may be pulled from remote data storage 204, utilizing the content address (414). Optionally, after being pulled from the remote data storage, the data block may be written to the local data log 208 and the local data map may be updated accordingly (416). This may allow the data block to be accessed locally for future reads.

The methods and systems described herein may provide several advantages. For example, as described above, data back-up may be facilitated. The remote data storage 204 may serve as back-up storage. Because the client device 205 automatically uploads changes to data blocks to the remote data storage 204, the back-up is not overly burdensome on users of the client device 205 and does not require extra diligence on the part of the users. In various embodiments, the remote data storage 204 may be ordinarily inaccessible to the client device 205. In these embodiments, a user of the client device may affirmatively log into to remote data storage 204 to perform a back-up.

The methods and systems described herein may also promote device accessibility. For example, the remote block map may correspond to a particular client device 205. Accordingly, a user of the client device 205 may log into the remote data storage 204 on a new device, access the remote block map, and re-create the client device 205 on the new device. With access to the block map and functionality for implementing the methods and systems above, the new device may boot directly from the remote storage 204 to implement the device. In embodiments where other data is present on the new device, functionality may be provided to hash and log this data to form a local data map. Because many data blocks are common across different devices that run similar operating systems and applications, this may minimize the number of data blocks that must be pulled from the remote data storage 204. To implement this functionality at a new

device, a user may be provided with a USB drive or other storage device comprising, for example, a version storage driver 108, authentication credentials to the remote storage device 204 and/or a block map corresponding to the remote block map. The ability to re-create the client device 205 on a new machine may provide a number of benefits. For example, in the event of the loss of a client device 205, a clone of the device could be created on a new device by simply implementing the storage driver 108 and accessing the remote block map. Also, for example, a user may be able to access their client device 205 while traveling without having to physically transport the device.

Various other advantages of the disclosed systems and methods arise from the fact that client device 205 data is present at the remote data storage 204. For example, data at the remote data store 204 may be scanned for viruses. Because any viruses that are present would be executing at the client device 205 and not at the remote data store 204, it may be difficult for a virus to hide its existence at the remote data store 204. Data blocks at the remote data store 204 that are found to include a virus signature may be deleted and/or flagged as potentially infected.

Still other advantages of the disclosed systems and methods arise from embodiments where an enterprise stores data from many client devices 205 at a single remote data store 204. For example, each individual client device 205 may have a unique remote block map stored at remote block map log 210. The remote data section 212 of the remote data store 204 may be common to all client devices 205 of the enterprise (*e.g.*, computer devices on a company's network, mobile phones on a mobile carrier's network, *etc.*). Because many data blocks are common on similar computer devices, implementing a common remote data section 212 may save significant storage space. In addition, enterprise administrators may be able to update applications on some or all of the client devices 205 by updating or changing the appropriate data blocks 218 at the remote data section 212 and updating the remote block log for each client device 205. When each client device 205 re-authenticates itself to the remote data storage 204, the changes to the block log may be downloaded, completing the update. Also, when remote data from multiple client devices 205 is commingled, processing required to perform virus checking may be significantly reduced because duplicated data blocks may only need to be scanned once.

It will be appreciated that a client device 205 may be any suitable type of computing device including, for example, desktop computers, laptop computers, mobile phones, palm top computers, personal digital assistants (PDA's), *etc.* As used herein, a "computer," "computer system," "computer device," or "computing device," may be, for example and without

limitation, either alone or in combination, a personal computer (PC), server-based computer, main frame, server, microcomputer, minicomputer, laptop, personal data assistant (PDA), cellular phone, pager, processor, including wireless and/or wireline varieties thereof, and/or any other computerized device capable of configuration for processing data for standalone application and/or over a networked medium or media. Computers and computer systems disclosed herein may include operatively associated memory for storing certain software applications used in obtaining, processing, storing and/or communicating data. It can be appreciated that such memory can be internal, external, remote or local with respect to its operatively associated computer or computer system. Memory may also include any means for storing software or other instructions including, for example and without limitation, a hard disk, an optical disk, floppy disk, ROM (read only memory), RAM (random access memory), PROM (programmable ROM), EEPROM (extended erasable PROM), and/or other like computer-readable media.

The term "computer-readable medium" as used herein may include, for example, magnetic and optical memory devices such as diskettes, compact discs of both read-only and writeable varieties, optical disk drives, and hard disk drives. A computer-readable medium may also include memory storage that can be physical, virtual, permanent, temporary, semi-permanent and/or semi-temporary.

It is to be understood that the figures and descriptions of embodiments of the present invention have been simplified to illustrate elements that are relevant for a clear understanding of the present invention, while eliminating, for purposes of clarity, other elements, such as, for example, details of system architecture. Those of ordinary skill in the art will recognize that these and other elements may be desirable for practice of various aspects of the present embodiments. However, because such elements are well known in the art, and because they do not facilitate a better understanding of the present invention, a discussion of such elements is not provided herein.

It can be appreciated that, in some embodiments of the present methods and systems disclosed herein, a single component can be replaced by multiple components, and multiple components replaced by a single component, to perform a given function or functions. Except where such substitution would not be operative to practice the present methods and systems, such substitution is within the scope of the present invention. Examples presented herein, including operational examples, are intended to illustrate potential implementations of the present method and system embodiments. It can be appreciated that such examples are intended

primarily for purposes of illustration. No particular aspect or aspects of the example method, product, computer-readable media, and/or system embodiments described herein are intended to limit the scope of the present invention.

It should be appreciated that figures presented herein are intended for illustrative purposes and are not intended as design drawings. Omitted details and modifications or alternative embodiments are within the purview of persons of ordinary skill in the art. Furthermore, whereas particular embodiments of the invention have been described herein for the purpose of illustrating the invention and not for the purpose of limiting the same, it will be appreciated by those of ordinary skill in the art that numerous variations of the details, materials and arrangement of parts/elements/steps/functions may be made within the principle and scope of the invention without departing from the invention as described in the appended claims.

CLAIMS

We claim:

1. A system for remote storage of data, the system comprising:

a processor circuit comprising at least one processor;

a local data storage device in electronic communication with the processor circuit, wherein the local data storage device comprises:

a local block map, wherein the local block map comprises a plurality of mappings, wherein each mapping maps an identifier of a data block to a corresponding content address; and

a log-structured local data storage comprising data units organized by content address; and

a memory circuit operatively associated with the processor circuit, wherein the memory circuit comprises instructions that, when executed by the processor circuit, cause the processor circuit to:

receive an electronic write request from an application, wherein the write request comprises an identifier of a first data block and a value for the first data block;

derive a content address of the first data block considering the value for the first data block;

write a mapping to a logical end of the local block map, wherein the mapping maps the identifier of the first data block to the content address;

write the mapping to a remote block map;

determine if the content address is present at the local data storage;

conditioned upon the content address not being present at the local data storage:

write the value of the first data block to the local storage at a first location; and

write to the local storage metadata associating the content address with the first location.

2. The system of claim 1, wherein the plurality of mappings are logically arranged in the local block map in chronological order based on when each mapping was written to the local block map.

3. The system of claim 1, wherein deriving the content address comprises applying a hash algorithm to the value for the first data block.

4. The system of claim 3, wherein the hash algorithm is selected from the group consisting of SHA-0, SHA-1, SHA-2, SHA-3 and MD5.

5. The system of claim 1, wherein the first data block is at least one size selected from the group consisting of 512 bytes, 520 bytes, 1024 bytes, 2048 bytes and 4096 bytes.

6. The system of claim 1, wherein the local block map is organized according to a log-structured format.

7. The system of claim 1, wherein the remote block map is organized according to a log-structured format.

8. The system of claim 1, further comprising marking the mapping as un-pushed when a remote data storage comprising the remote block data map is unavailable.

9. The system of claim 1, wherein the memory circuit further comprises instructions that, when executed by the processor circuit, cause the processor circuit to, conditioned upon the content address not being present at the local data storage:

determine whether the content address is present at a remote storage;

write the value of the first data block to the remote storage at a first location; and

write to the remote storage metadata associating the content address with the first location.

10. A method for remote storage of data, the method comprising:

receiving an electronic write request from an application, wherein the write request comprises an identifier of a first data block and a value for the first data block;

deriving a content address of the first data block considering the value for the first data block;

writing a mapping to a logical end of a local block map, wherein the mapping maps the identifier of the first data block to the content address, wherein the local block map comprises a plurality of mappings, wherein each of the plurality of mappings maps an identifier of a data block to a corresponding content address;

writing the mapping to a remote block map;

determining if the content address is present at a local data storage, wherein the local data storage is log-structured and comprises data units organized by content address;

conditioned upon the content address not being present at the local data storage:

writing the value of the first data block to the local storage at a first location; and

writing to the local storage metadata associating the content address with the first location.

11. A portable data storage device for re-creating a client device on a computer machine, the device comprising a computer readable medium having written thereon:

a local block map, wherein the local block map comprises a plurality of mappings, wherein each mapping maps an identifier of a data block to a corresponding content address;

a log-structured local data storage comprising data units organized by content address; and

instructions that, when executed by a processor circuit, cause the processor circuit to:

receive an electronic write request from an application, wherein the write request comprises an identifier of a first data block and a value for the first data block;

derive a content address of the first data block considering the value for the first data block;

write a mapping to a logical end of the local block map, wherein the mapping maps the identifier of the first data block with the content address;

write the mapping to a remote block map;

determine if the content address is present at the local data storage;

conditioned upon the content address not being present at the local data storage:

write the value of the first data block to the local storage at a first location; and

write to the local storage metadata associating the content address with the first location.

12. A computer readable medium comprising instructions thereon that, when executed by at least one processor, cause the at least one processor to:

upon receipt of a write request comprising an identifier of a data block and a value of the data block, derive a content address for the data block based on the value of the data block;

update a local block map to associate the identifier with the content address;

update a remote block map to associate the identifier with the content address;

determine whether a log-structured local data log comprises the content address;

conditioned upon the local data log not comprising the content address:

write the value of the data block to the local data log at a first location; and

write to the local data log metadata associating the content address with the first location;

determine whether a remote data log comprises the content address;

conditioned upon the remote data log not comprising the content address:

write the value of the data block to the remote data log at a first remote location; and

write to the remote data log metadata associating the content address with the first remote location.

13. The computer readable medium of claim 12, wherein the remote data log is log-structured.

14. The computer readable medium of claim 12, wherein the remote data log is organized according to at least one of a hierarchal storage structure and an indexed storage structure.

15. The computer readable medium of claim 12, wherein updating the local block map comprises writing a mapping to a logical end of the local block log, wherein the mapping maps the identifier of the data block with the content address.

16. The computer readable medium of claim 12, wherein updating the remote block map comprises writing a mapping to a logical end of the remote block log, wherein the mapping maps the identifier of the data block with the content address.

17. A computer system comprising:

a processor circuit comprising at least one processor;

a local data storage device in electronic communication with the processor circuit, wherein the local data storage device comprises:

a local block map, wherein the local block map comprises a plurality of mappings, wherein each mapping maps an identifier of a data block to a corresponding content address; and

a log-structured local data storage comprising data units organized by content address; and

a memory circuit operatively associated with the processor circuit, wherein the memory circuit comprises instructions that, when executed by the processor circuit, cause the processor circuit to:

receive an electronic read request from an application, wherein the read request comprises an identifier of a first data block;

determine if the local block map comprises a content address associated with the identifier of the first data block;

conditioned upon the local block map comprising a content address associated with the identifier of the first data block, retrieving the content address from the local block map;

conditioned upon the local block map not comprising the content address, retrieving the content address from a remote block map;

determine whether the content address appears in the local data storage;

conditioned upon the content address appearing in the local data storage, retrieving a value associated with the content address in the local storage and returning the value to the application as a value for the first data block; and

conditioned upon the content address not appearing in the local data storage, retrieving a value associated with the content address in the remote storage and returning the value to the application as an identifier of a value for the first data block.

18. The system of claim 17, wherein the memory circuit comprises instructions that, when executed by the processor circuit, cause the processor circuit to, conditioned upon the content address not appearing in the local data storage, write the value associated with the content address in the remote storage to the local data storage.

19. The system of claim 17, wherein the plurality of mappings are logically arranged in the local block map in chronological order based on when each mapping was written to the local block map.

20. A computer-implemented method comprising:

receiving by a processor circuit an electronic read request from an application, wherein the read request comprises an identifier of a first data block, and wherein the processor circuit comprises at least one processor and is in communication with a local data storage;

determining by the processor circuit if a local block map at the local data storage comprises a content address associated with the identifier of the first data block;

conditioned upon the local block map comprising a content address associated with the identifier of the first data block, retrieving the content address from the local block map by the processor circuit;

conditioned upon the local block map not comprising the content address, retrieving the content address from a remote block map by the processor circuit;

determining by the processor circuit whether the content address appears in the local data storage;

conditioned upon the content address appearing in the local data storage, retrieving by the processor circuit a value associated with the content address in the local data storage and returning the value to the application as a value for the first data block; and

conditioned upon the content address not appearing in the local data storage, retrieving a value associated with the content address in the remote storage and returning the value to the application as an identifier of a value for the first data block.

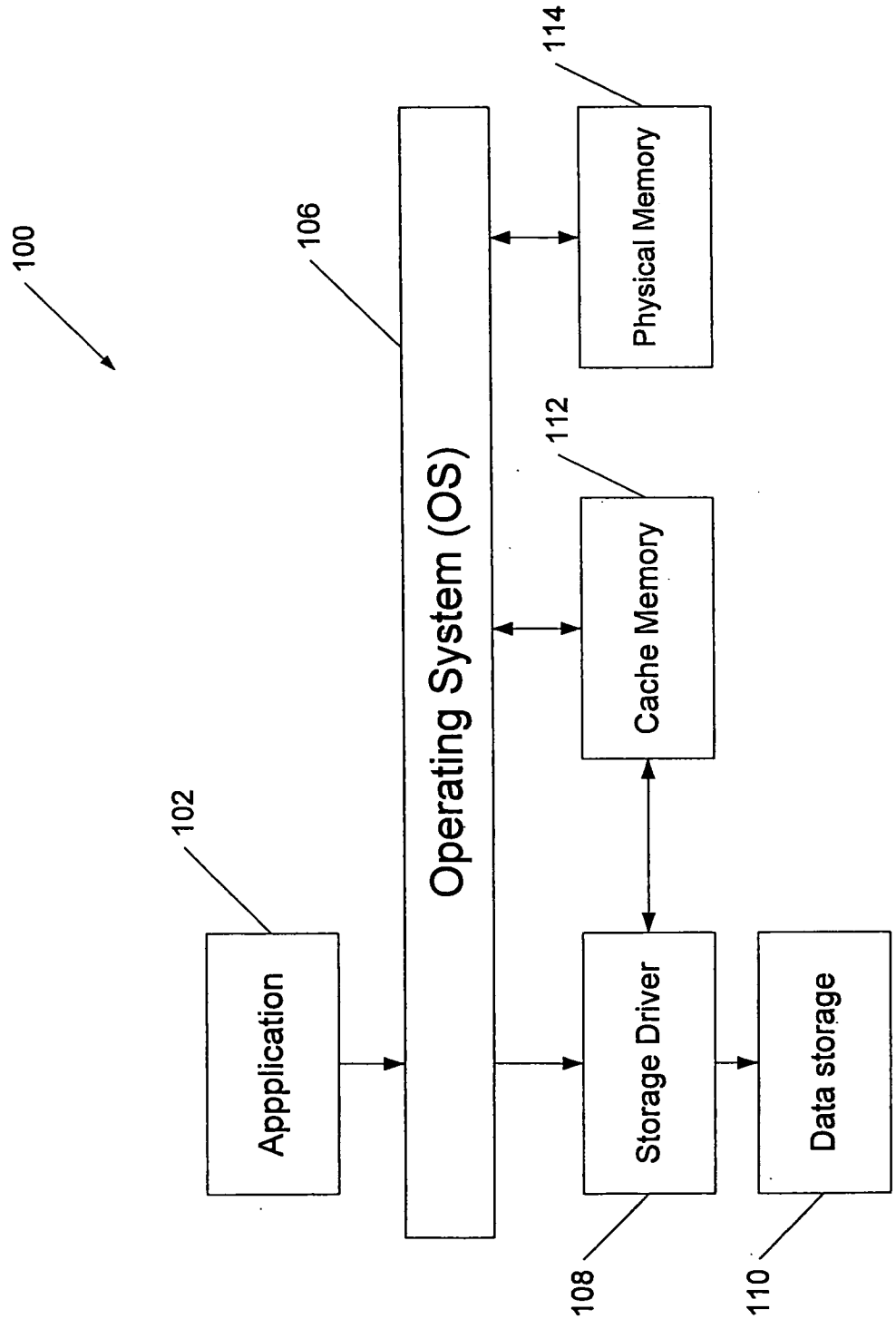
100
102
Appplication
106
Operating System (OS)
108
Storage Driver
110
Data storage
112
Cache Memory
114
Physical Memory

Figure 1

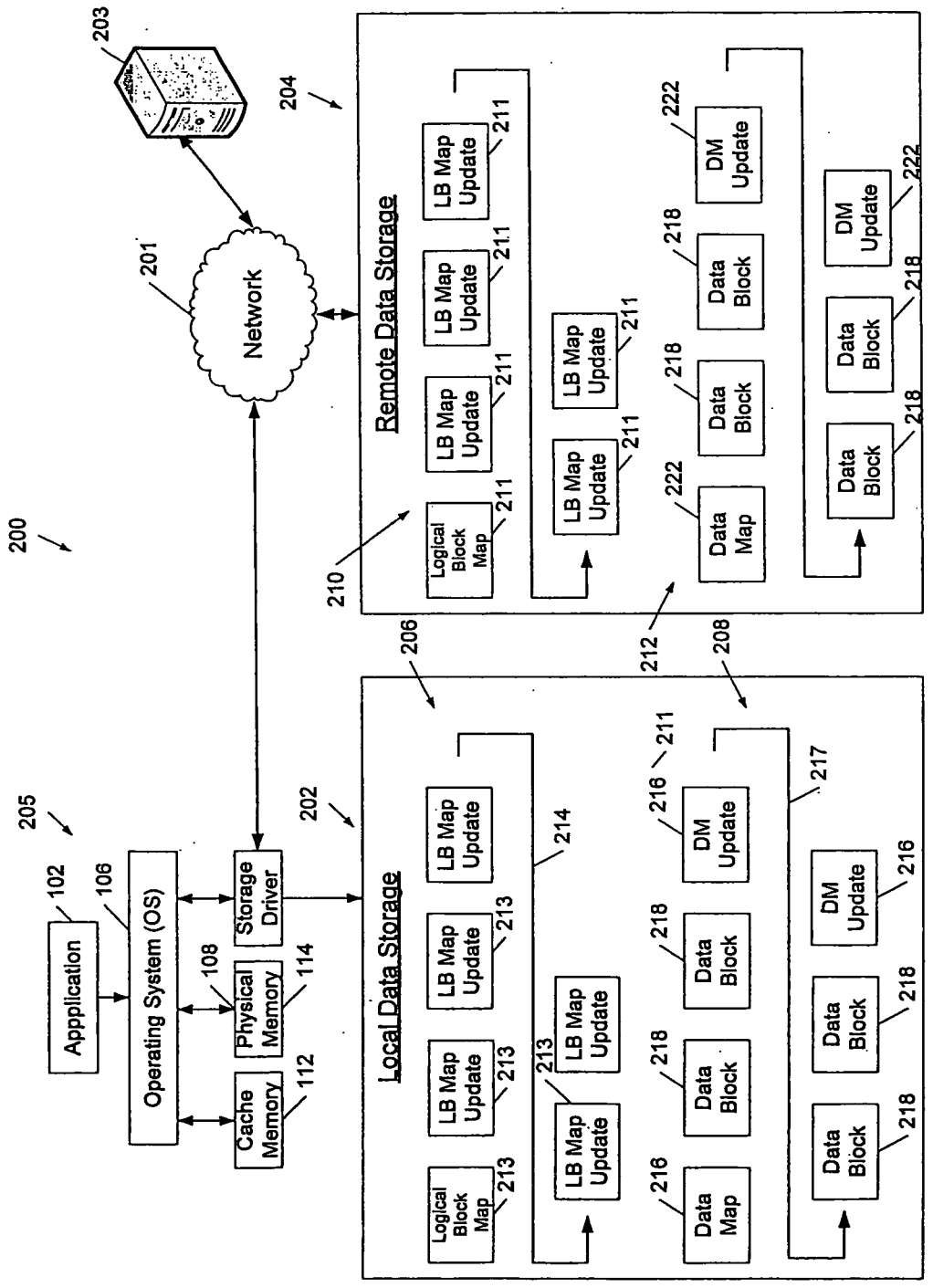
200
205
Appplication
102
Operating System (OS)
106
108
Physical Memory
114
Cache Memory
112
Storage Driver
202
Network
201
203
204
Remote Data Storage
210
Logical Block Map
211
LB Map Update
Data Map
222
Data Block
218
DM Update
206
Local Data Storage
212
208
213
214
216
217

Figure 2

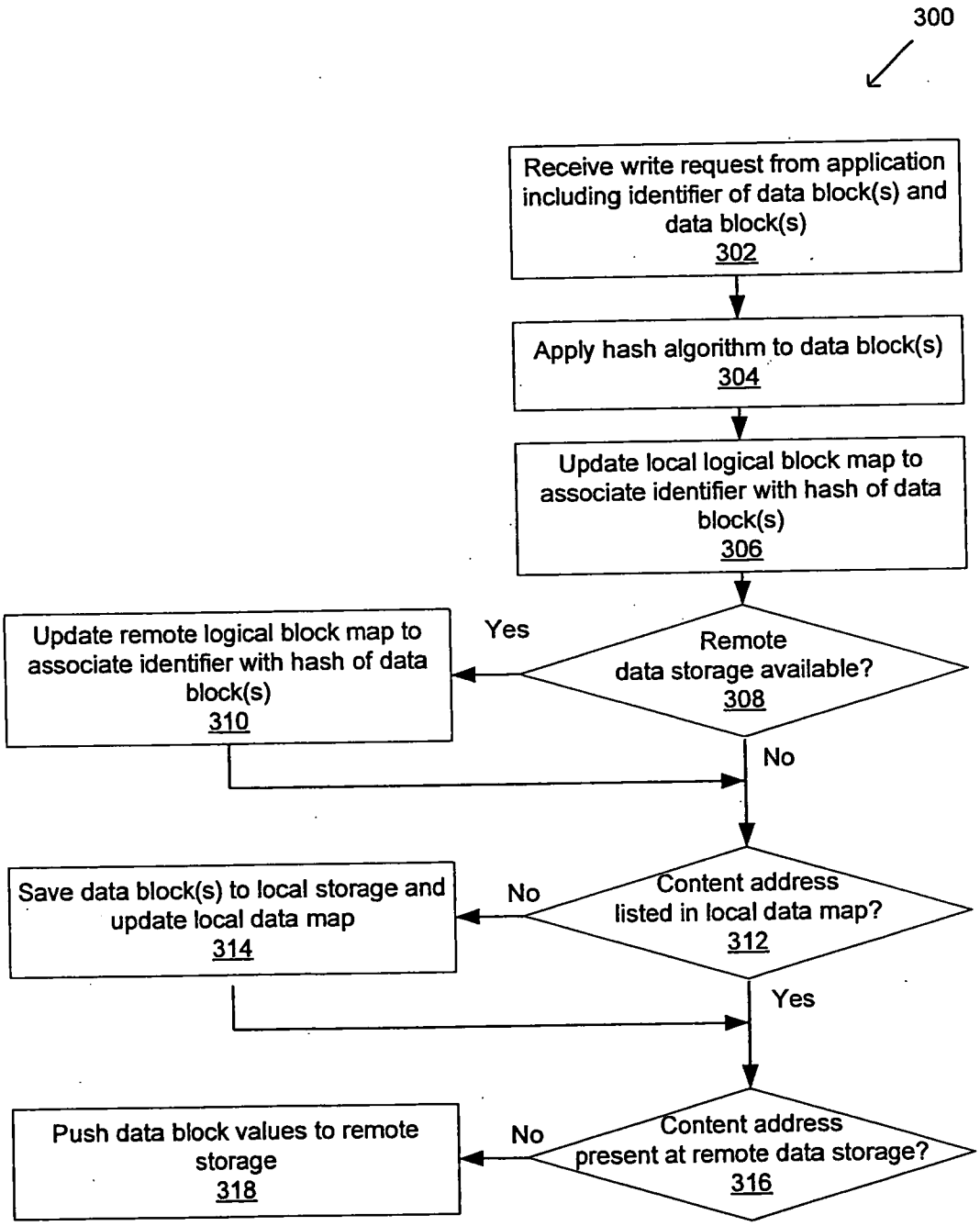
300
Receive write request from application including identifier of data block(s) and data block(s)
302
Apply hash algorithm to data block(s)
304
Update local logical block map to associate identifier with hash of data block(s)
306
Remote data storage available?
308
Yes
Update remote logical block map to associate identifier with hash of data block(s)
310
No
Content address listed in local data map?
312
No
Save data block(s) to local storage and update local data map
314
Yes
Content address present at remote data storage?
316
No
Push data block values to remote storage
318

Figure 3

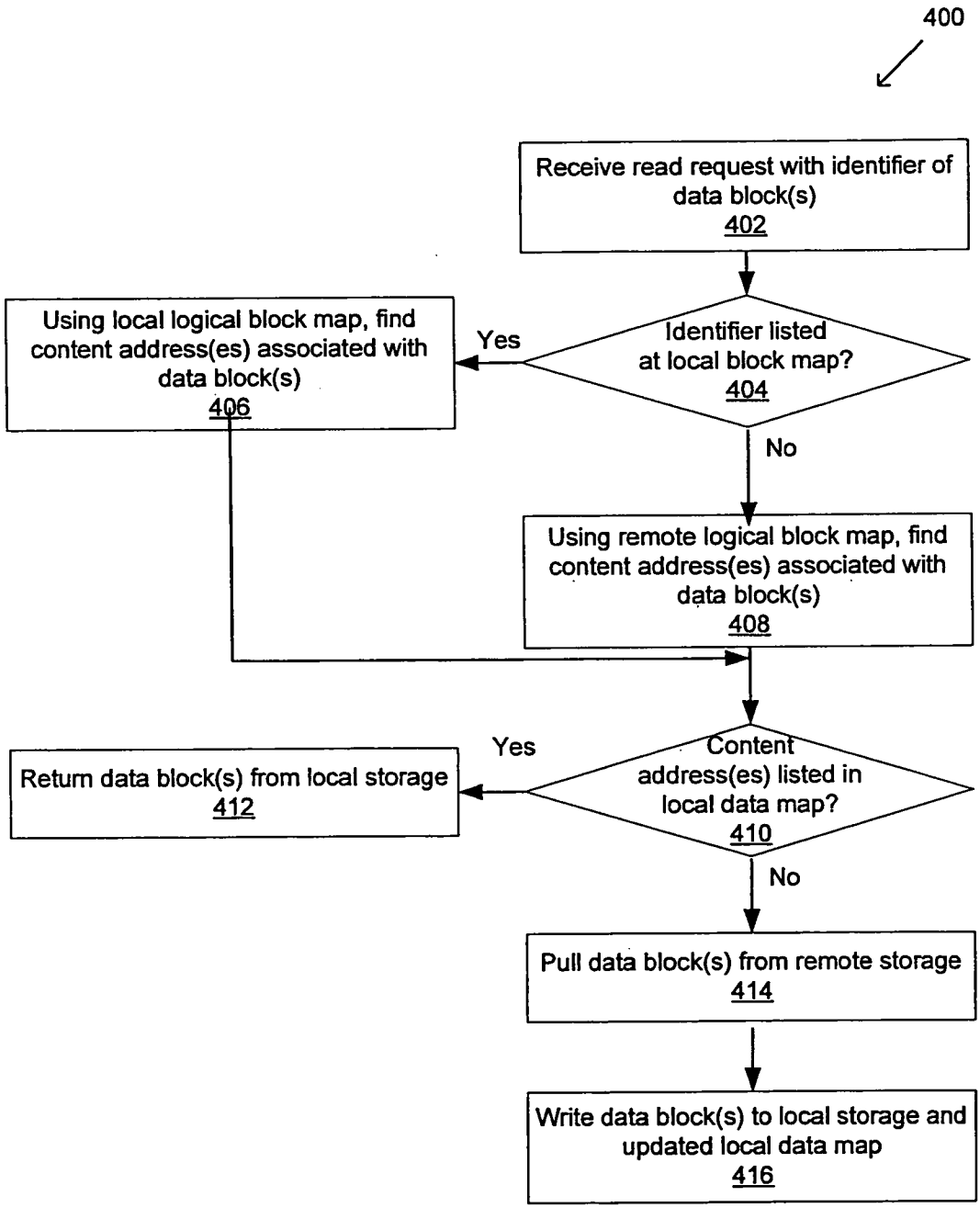
400
Receive read request with identifier of data block(s)
402
Identifier listed at local block map?
404
Yes
Using local logical block map, find content address(es) associated with data block(s)
406
No
Using remote logical block map, find content address(es) associated with data block(s)
408
Content address(es) listed in local data map?
410
Yes
Return data block(s) from local storage
412
No
Pull data block(s) from remote storage
414
Write data block(s) to local storage and updated local data map
416

Figure 4

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2010/000317

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06F12/08 G06F3/06
ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2004/225837 A1 (LEWIS RUSSELL L [US]) 11 November 2004 (2004-11-11) paragraphs [0010] - [0017], [0023] - [0026]; figures 1,2	1-20
A	BRESSOUD T C ET AL: "OPEN CAS: A FLEXIBLE ARCHITECTURE FOR CONTENT ADDRESSABLE STORAGE" PROCEEDINGS OF THE ISCA INTERNATIONAL CONFERENCE, PARALLEL ANDDISTRIBUTED COMPUTING SYSTEMS, XX, XX, 15 September 2004 (2004-09-15), pages 580-587, XP009068171 page 580, left-hand column page 585, right-hand column, line 22 - page 586, right-hand column, line 22	1-20

-/--

[X] Further documents are listed in the continuation of Box C.

[X] See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
9 July 2010	21/07/2010

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Nielsen, Ole

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2010/000317

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2006/174156 A1 (BALASUBRAMANIAN SRIDHAR [US]) 3 August 2006 (2006-08-03) paragraphs [0007] - [0010]	1-20
A	US 6 804 718 B1 (PANG HWEE HWA [SG] ET AL) 12 October 2004 (2004-10-12) column 3, line 51 - column 4, line 16	1-20

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/US2010/000317

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2004225837 A1	11-11-2004	NONE	
US 2006174156 A1	03-08-2006	NONE	
US 6804718 B1	12-10-2004	NONE	